

[illegible][illegible]

[illegible]

COPYRIGHT (c) 1978, 1980, 1982, 1984 BY
DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
ALL RIGHTS RESERVED.

THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
TRANSFERRED.

THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
CORPORATION.

DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.

✦ ✦

Revision history:

```

ON HISTORY:
V03-035 MIR0410      Michael I. Rosenblum      11-Apr-1984
      Make Unit init macro store the default parity setting
      in the ucb from the system location.

V03-034 MIR0320      Michael I. Rosenblum      15-Mar-1984
      Add missing label to CLASS_CTRL_INIT generate all others
      Propagate GLOBAL flag thru $TTYDEFS to get symbols defined
      in SYS.STB.

V03-033 MIR0070      Michael I. Rosenblum      13-jul-1983
      Add macros CLASS_UNIT_INIT and CLASS_CTRL_INIT.

V03-032 JLV0032      Jake Vannoy
      Create TTYMACS.MAR and TTYDEF.SDL. Move $TTYDEFS and $TTYMODEM
      and $TTYDEF to TTYDEF.SDL

V03-031 RKS0031      RICK SPITZ      18-APR-1983
      ADD NEW INTERNAL FUNCTION DEFINITIONS TO ALLOW
      MORE EFFICIENT STARTIO DISPATCHING OF FUNCTIONS.
      ADD DEFINITIONS FOR PORT AUTOXOFF FEATURE.

V03-030 MIR0031      Michael I. Rosenblum      1-Apr-1983
      Add bits to enable the user to specify frame size, and non-interruptable
      multiecho's. Add in alternate echo string data structures.

V03-029 MIR0029      Michael I. Rosenblum      21-Mar-1983
      Add bit to allow a standard terminator set to be ignored

```


and insert/overstrike toggle.

V03-028 RKS0028 RICK SPITZ 14-MAR-1982
RENAME REDUCB FIELD TO PHYUCB

V03-027 MIR1026 Michael I. Rosenblum 01-Mar-1983
Add broadcast class quad word, fields to allow multiecho
to be one level recursive, space for a backspace count, and
an area in the typeahead buffer to handle recall of the last
command.

V03-026 MIR0026 Michael I. Rosenblum 11-Feb-1983
Add locations to point to dispatch tables for input
character dispatching.

V03-025 RKS0023 RICK SPITZ 05-FEB-1983
ADD DEFINITIONS TO SUPPORT DMA IN YCPORT LEVEL

V03-024 MIR0025 Michael I. Rosenblum 1-Feb-1983
add modifiers long word to the terminal read buffer
structure. Add item list definitions.

V03-023 MIR0024 Michael I. Rosenblum 26-Jan-1983
Change read packet definition, to allow more flexibility
and a clean implimentation of both input editing and
read with verification.

V03-022 MIR0022 Michael I. Rosenblum 19-Jan-1983
add UCBSW IT UNITBIT Word that contains one bit set
to indicate which unit this line is, used by modem
control and DZ11, must be set for all controllers.
Merge CRB and IDB definitions into the main system
definition file.

V03-021 RKS0021 RICK SPITZ 13-JAN-1983
REPAIR PROBLEM WITH PORT VECTOR MACRO

V03-020 MIR0018 Michael I. Rosenblum 07-Jan-1983
Add macro to build the port driver entry table. This
macro will allow us to rearrange the port table at our
discretion and only require assembling and relinking
of the port driver.

V03-019 MIR0017 Michael I. Rosenblum 04-Jan-1983
Add power fail bit to the unit state vector and
class powerfail callback. This will
allow us to make the power fail checks in the class driver
in only one place.

V03-018 MIR0016 Micheal I. Rosenblum 29-Dec-1982
Add TIMSET macro and TTY\$V_PC_NOTIME bit.
The TIMSET macro should take care of all the places
where duetim and the TIM bit are normaly set.

V03-017 MIR0015 Michael I. Rosenblum 21-Dec-1982
Add CLASS_DISCONNECT, CLASS_FORK, and PORT_FORK vector

entry points.
Add FD fork dispatch bit table.

V03-016 MIR0014 Michael I. Rosenblum 17-Dec-1982
change xon and xoff bits in tty\$b_tt_tank to reflect
the change of functionality in the xon and xoff port
driver functions.

V03-015 MIR0013 Michael I. Rosenblum 14-Dec-1982
Split TTYDEFS into the following sections:

TTLOGDEF	The logical terminal UCB extensions
TTCLSDEF	The terminal class driver and port driver independant extensions
TTPRTDEF	The terminal port driver dependent region, _TP_.

V03-014 MIR0011 Michael I. Rosenblum 18-Nov-1982
Change definition by removing it from the first state longword
for CTRLR to indicate that the prompt string is being clocked
when this bit is set.
Add EDITREAD state bit to the first longword. This bit indicates
that a read editing string is being output.
Add SKIPCRLF to the second state longword. This bit indicates
that the linefeed following a CR in the beginning of the
prompt string is to be skipped.

V03-013 MIR0010 Michael I. Rosenblum 09-Nov-1982
Add definitions as follows:

TTY\$\$_RB_TERM	Address of the terminator mask (either standard mask or new mask in the read packet).
TTY\$\$_RB_PRMP	Offset from the beginning of the read packet to the end of the prompt string.
TTY\$\$_TA_INAHD	The number of characters currently in typeahead buffer.

V03-012 RKS0012 RICK SPITZ 05-APR-1982
CONVERT SPARE BYTE INTO CURRENT OUTPUT ESCAPE RULE

V03-011 RKS0011 RICK SPITZ 11-JAN-1981
ADD EXTENSION REGION FOR READ BUFFER HEADER

V03-010 RKS0010 RICK SPITZ 15-DEC-1981
ADD NEW CONTROL DEFINITION FOR CHARACTER TYPE TABLE.
MOVE LOWER CASE DEFINITION BIT FOR TYPE TABLE.
ADD ALTERNATE DRIVER LOCATION IN UCB.
ADD MAP,ALTLEN,SPARE IN UCB.
MOVE LINE DISABLE BIT IN MAINT FIELD.
ADD IDB FIELD DEFINITION FOR DZ32.

V03-009 BLS0116 Benn Schreiber 2-Dec-1981
Correct IF_NOT_STATE macro

V03-008 JLV0125 Jake VanNoy 28-Oct-1981
Add NOSET, NOCLEAR, NOMOD, PRIV TO MOD to \$TTYMACS.
Also, add one bit checking to IF_STATE and IF_NOT_STATE.

V03-007 JLV0103 Jake VanNoy 28-Oct-1981
Changed TTYDEFS to \$TTYDEFS. Move \$TTYDEF from SYSDEF
to this module.

V03-006 PHL0020 Peter H. Lipman 27-Oct-1981
Moved TT_DEVDP1 to fixed portion of UCB. It is a
synonym for the new DEVDEPND2 cell.

V03-005 JLV0089 Jake VanNoy 9-Sep-1981
Added AUTOP, autobaud pending timeout.

V03-004 RKS004 RICK SPITZ 26-AUG-1981
MOVE RDUE EARLIER IN UCB TO ALLOW EASIER EXTENSION OF UCB

V03-003 RKS0003 RICK SPITZ 20-AUG-1981
ADD NEW STATE BIT DEFINITIONS
ADD OUT OF BAND SUMMARY MASK AND QUE HEAD.

V03-002 RKS0002 Rick Spitz 27-JULY-1981
Restructure device dependent portion of UCB and
add new UCB fields for modem control, split speed
and output optimizations
Redefine IF_STATE, GTSBITS Macros to allow quadword
state field.
Add modem definitions for DEC052 modem control

V03-001 RKS0001 RICK SPITZ 13-NOV-1980
Revise Ucb extensions for terminal driver


```

:
:
:  TERMINAL DRIVER MACROS
:

```

```

:  THESE MACROS ARE USED TO GENERATE CODE FOR IF STATE MACROS.
:  THEY GENERATE A SEQUENCE OF ONE OR TWO BIT(?) 7 BRANCH
:  OR BIS(?) INSTRUCTION COMBINATIONS DEPENDING ON THE SEPERATION OF THE
:  BITS BEING TESTED IN THE STATE QUADWORD.

```

```

.MACRO $TTYMACS

```

```

.MACRO GTSBITS BITS,MODE,TARGET,BRANCH,?L1

```

```

F=0
Z0=3
X0=0
W0=0
Z1=3
X1=0
W1=0
.IRP Y,<BITS>
T=TTY$V_SX-'Y
.IF LE 32-T
X1=T-32-3
.IF LT X1-Z1
Z1=X1
.ENDC
W1=<TTY$M_ST-'Y>!W1
.ENDC
.IF GT 32-T
X0=T-3
.IF LT X0-Z0
Z0=X0
.ENDC
W0=<TTY$M_ST-'Y>!W0
.ENDC
.ENDR
.IF NE W0
GTSBITS1 Z0,W0,MODE,0
.IF NB TARGET
.IF IDN BRANCH,BEQL
.IF NE W1
F=1
BNEQ L1
.IFF
BEQL TARGET
.ENDC
.ENDC
.IF DIF BRANCH,BEQL
BNEQ TARGET
.ENDC
.ENDC
.ENDC
.IF NE W1
GTSBITS1 Z1,W1,MODE,4
.IF NB TARGET
BRANCH TARGET

```



```
L1: .ENDC
    .ENDC
    .IF      NE      F
    .ENDC
    .ENDM      GTSBITS

    .MACRO      GTSBITS1      Z,WX,MODE,BIAS
    WX=WX@-<Z*8>
    X=WX@-8
    .IF      EQ      X
    BI'MODE'B      #WX,BIAS+Z(R2)
    .IFF
    X=WX@-16
    .IF      EQ      X
    BI'MODE'W      #WX,BIAS+Z(R2)
    .IFF
    BI'MODE'L      #WX,BIAS+Z(R2)
    .ENDC
    .ENDC
    .ENDM      GTSBITS1

    .MACRO      SET_STATE      NAME
    GTSBITS <NAME>,S
    .ENDM      SET_STATE

    .MACRO      CLR_STATE      NAME
    GTSBITS <NAME>,C
    .ENDM      CLR_STATE

    .MACRO      IF_STATE      NAME,TARGET
    CNT = 0
    .IRP      Y,<NAME>
    CNT = CNT + 1
    .ENDR

    .IF EQUAL CNT - 1
    ONE_BIT <NAME>,S,TARGET
    .IFF
    GTSBITS <NAME>,T,TARGET,BNEQ
    .ENDC
    .ENDM      IF_STATE

    .MACRO      IF_NOT_STATE      NAME,TARGET
    CNT = 0
    .IRP      Y,<NAME>
    CNT = CNT + 1
    .ENDR

    .IF EQUAL CNT - 1
    ONE_BIT <NAME>,C,TARGET
    .IFF
    GTSBITS <NAME>,T,TARGET,BEQL
    .ENDC
```

```
.ENDM IF_NOT_STATE
```

```
.MACRO ONE_BIT BIT,BRANCH,TARGET
BB'BRANCH' #TTY$V_SX_'BIT',(R2),'TARGET'
.ENDM ONE_BIT
```

```
:: Bit checking for setmode/char changes to DEVDEPND2. Assumes
:: R0 = input, R1 = Bits changing, R3 = IRP, R5 = UCB.
::
```

```
.MACRO NOSET BIT,?L1
BBC #TT2$V_'BIT',R1,L1
BICL #TT2$M_'BIT',R0
```

L1:

```
.ENDM NOSET
```

```
.MACRO NOCLEAR BIT,?L1
BBC #TT2$V_'BIT',R1,L1
BISL #TT2$M_'BIT',R0
```

L1:

```
.ENDM NOCLEAR
```

```
.MACRO NOMOD BIT,?L1
BBC #TT2$V_'BIT',R1,L1
XORL2 #TT2$M_'BIT',R0
```

L1:

```
.ENDM NOMOD
```

```
.MACRO PRIV_TO_MOD BIT,ERROR = NOPRIV_EXIT,?L1
BBC #TT2$V_'BIT',R1,L1
BITL #<<1@PRV$V LOG IO>!--
<1@PRV$V PRY IO>>,-
@IRP$L_ARB(R3)
```

```
BNEQ L1
BRW 'ERROR'
```

L1:

```
.ENDM PRIV_TO_MOD
```



```

TIMSET - macro to handle setting timeout's
++
TIMSET
Description:
This macro handles all of the timesetting needs of the terminal
driver. It will check the port control word before any action to determine
if timeouts are required for this device.
Inputs:
LEN = location containing the length of the string
      OR if WORK is blank
      The minimum number of seconds to wait.
WORK = Temp register. If blank alternate form of this macro
      is generated to wait for a constant time
INTEXP = if not blank then the interrupt expected bit is set
--
.MACRO TIMSET LEN,WORK,INTEXP,?L1,?L2
.L1:
.NB INTEXP
BBC      #TTY$V_PC_NOTIME,UCBSW_TT_PRTCTL(R5),L1
BBCS     #UCBSW_INT,UCBSW_STS(R5),L2
BRB      L2
.IFF
BBS      #TTY$V_PC_NOTIME,UCBSW_TT_PRTCTL(R5),L2
.ENDC
L1:
.IF
B WORK
ADDL3    #LEN+1,G^EXESGL_ABSTIM,UCBSL_DUETIM(R5) ; SET TIME OUT
.IFF
DIVL3    #4,LEN,WORK
ADDL     #2,WORK
ADDL3    WORK,G^EXESGL_ABSTIM,-
          UCBSL_DUETIM(R5)
.ENDC
.IF NB INTEXP
BISB     #UCBSM_INT!UCBSM_TIM,-
          UCBSW_STS(R5)
.IFF
BISB     #UCBSM_TIM,-
          UCBSW_STS(R5)
.ENDC
L2:
.ENDM

```

```
: $VECINI - Macro to start the port vector table
: ++
: $VECINI
: Description:
:
: This macro will generate a port entry vector table and a
: null entry point for a port driver. Initially this table will be
: filled with calls to the null entry point and filled in by later
: calls to the $VEC macro. This macro generates the $VEC macro and
: the $VECEND macro
:
: Inputs:
:
: DRIVERNAME = The two letter driver prefix
: PREFIX = (Optional) Prefix to be added to the symbols in later
: calls to $VEC. Defaulted to PORT_.
:
: Implicit Inputs:
:
: PREFIX_LENGTH = Number of bytes in the maximum size table.
:
: Generated labels:
: drivename$VEC = The start of the vector table
: drivename$VECEND = The end of the vector table
: drivename$NULL = Null entry point (RSB)
:
: --
: .MACRO $VECINI DRIVERNAME, NULL_ROUTINE, PREFIX=PORT_
: 'DRIVERNAME'$VEC:
: .REPEAT 'PREFIX'LENGTH/4
: .LONG NULL_ROUTINE
: ENDR
: 'DRIVERNAME'$VECEND:
```


\$VEC - Validates and generates vector table entry

++
\$VEC

Description:

This macro will validate and generate a vector table entry.
The position of this entry in the vector table may change from
version to version but the use of this macro will always generate
a working vector table or it will generate an error.

This macro call must follow a \$VECINI call.

Inputs:

ENTRY = The name of the table entry
ROUTINE = The name of the routine.

--
.MACRO \$VEC ENTRY,ROUTINE
.IF NDF PREFIX''ENTRY
.ERROR ; table location PREFIX''ENTRY undefined
.IFF
.=DRIVERNAME'\$VEC+PREFIX''ENTRY
.IF GE .-DRIVERNAME'\$VECEND
.ERROR ; Table location PREFIX''ENTRY out of range
.IFF
.LONG ROUTINE
.ENDC
.ENDC
.ENDM \$VEC

```
:      $VECEND - Generates the ending code for a vector table
:++
:$VECEND
:Description:
:      Will generate the vector tables null routine and
:      set the location counter to the correct place.
:Inputs:
:      END = Blank if this is the end of the table
:           non blank if the end of the table is not to be generated
:--
:      .MACRO $VECEND END
:      .='DRIVERNAME'$VECEND
:      .IIF BLANK, END, .LONG 0
:      .ENDM $VECEND
:      .ENDM $VECINI
```



```

:++
: Class_Ctrl_init - Macro to generate code controller init code common to all
:                   port drivers

```

```

: Description:

```

```

: This macro is provided to make sure that all the port drivers
: have a common set of controller-init code. This macro is required to
: be part of every terminal port driver's controller init code.

```

```

: Inputs:

```

```

: DPT - the symbolic name of the port's Driver prologue table
: VECTOR - The address of the port dispatch table generated with
:           The $VEC macro.

```

```

:--
: .macro CLASS_CTRL_INIT DPT,VECTOR,?L1,?L2,?L3,?L4
:
:   MOVL    G^TTY$GL DPT,R1      ; LOCATE CLASS DRIVER TO BIND TO
:   MOVZWL  DPT$W_VECTOR(R1),R0 ; GET ADDRESS OF CLASS VECTOR
:   ADDL3   R0,R1,R0            ; CALCULATE VIRTUAL ADDRESS
:
: ; RELOCATE CLASS VECTOR TABLE
:
: L1:
:   TSTL    (R0)                ; ALREADY RELOCATED OR DONE?
:   BLEQ    L2                  ; YES
:   ADDL    R1,(R0)+            ; ADD BIAS
:   BRB     L1                  ; LOOP TILL DONE
:
: ; RELOCATE PORT VECTOR TABLE
:
: L2:
:   MOVAL   DPT,R1
:   MOVAL   VECTOR,R0
:
: L3:
:   TSTL    (R0)                ; ALREADY RELOCATED OR DONE?
:   BLEQ    L4                  ; YES
:   ADDL    R1,(R0)+
:   BRB     L3
:
: L4:
:
:   .endm    class_ctrl_init

```

```

:++
: CLASS_UNIT_INIT - Macro that contains the code that the class driver
:                   needs in all of the port drivers unit init routines

```

```

: Description:

```

```

:   This macro provides a method of allowing the class driver
:   some code in every port drivers unit init routine. This routine
:   should be assumed black magic in the port driver as it's
:   contents and algorithms may be changed from release to release.

```

```

: Inputs:

```

```

:   R0 - contains the address of the port dispatch table for
:         This unit.

```

```

:--
: .macro CLASS_UNIT_INIT,?L1
:   MOVL G^TTY$GL_DPT,R1      ; ADDRESS OF CLASS DPT
:   MOVZWL DPT$W_VECTOR(R1),R2 ; LOCATE CLASS DRIVER VECTOR TABLE
:   ADDL R2,R1                ; RELOCATE BASE ADDRESS
:   MOVL R1,UCB$L_TT_CLASS(R5) ; SET TERMINAL CLASS DRIVER VECTOR
:
:   MOVL R0,UCB$L_TT_PORT(R5) ; SET PORT VECTOR TABLE ADDRESS UP
:   MOVL CLASS_GETNXT(R1),UCB$L_TT_GETNXT(R5)
:   MOVL CLASS_PUTNXT(R1),UCB$L_TT_PUTNXT(R5)
:   MOVL UCB$L_DDB(R5),R2      ; GET DDB ADDRESS
:   MOVL CLASS_DDT(R1),DDB$L_DDT(R2)
:   MOVL CLASS_DDT(R1),UCB$L_DDT(R5) ; SET DDT ADDRESS IN UCB
:
:   BBS #UCB$V_POWER,UCB$W_STS(R5),L1; IF THIS ISN'T POWER FAIL
:   MOVB G^TTY$GB_PARITY,UCB$B_TT_PARITY(R5); THEN SET THE DEFAULT
:   MOVB G^TTY$GB_PARITY,UCB$B_TT_DEPARI(R5); PARITY SETTINGS
:
: L1:
:
: .endm class_unit_init

```

;
; STO_TQE - Modem TQE macro

\$\$\$\$\$\$.MACRO STO_TQE OFFSET,SIZE,VALUE,BASE
= .
= 'SIZE OFFSET+BASE
= VALUE
= \$\$\$\$\$\$
.ENDM STO_TQE

.MACRO \$TTYMACS
.ENDM \$TTYMACS

.ENDM \$TTYMACS


```

.MACRO      $TTYDEFS,$GBL
$DEFINI    TTYDEFS,$GBL

$UCBDEF      $GBL
$CRBDEF      $GBL
$IDBDEF      $GBL
$TTYUCBDEF    $GBL
$TTYVECDDEF   $GBL
$TTYSYMDEF    $GBL
$TTYRBDEF     $GBL
$TTYISDEF     $GBL
$TTYILDEF     $GBL
$TTYTADDEF    $GBL

$DEFEND      TTYDEFS,$GBL,$DEF
.ENDM        $TTYDEFS

```

```
; DEFINE CRB,IDB OFFSETS

; UCB Extension
; Class and Port Vectors
; Misc Symbols
; Read Buffer
; Input Stack
; Itemlist Descriptor
; Typeahead buffer
```

DZ
VO

...
\$TTYMODEM exists here so that the name change to \$TTYMDMDEF
can happen without changing the driver. This should be cleaned
up when convenient.
...

.MACRO \$TTYMODEM \$GBL
\$DEFINI TTYMODEM,\$GBL

\$TTYMDMDEF ; Define equivalent name

\$DEFEND TTYMODEM,\$GBL,DEF
.ENDM \$TTYMODEM

.END

0402 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

